TA Mailinh

Gautier Manon

EFREI PARIS

L1 – Int 3

Promo : 2027

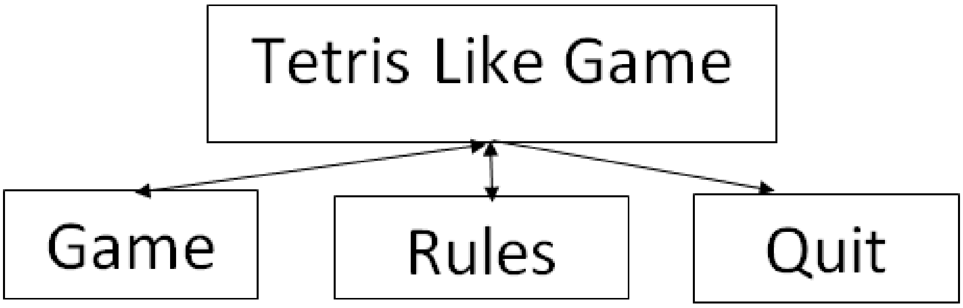
TETRIS GAME

2022 – 2023

Plan :

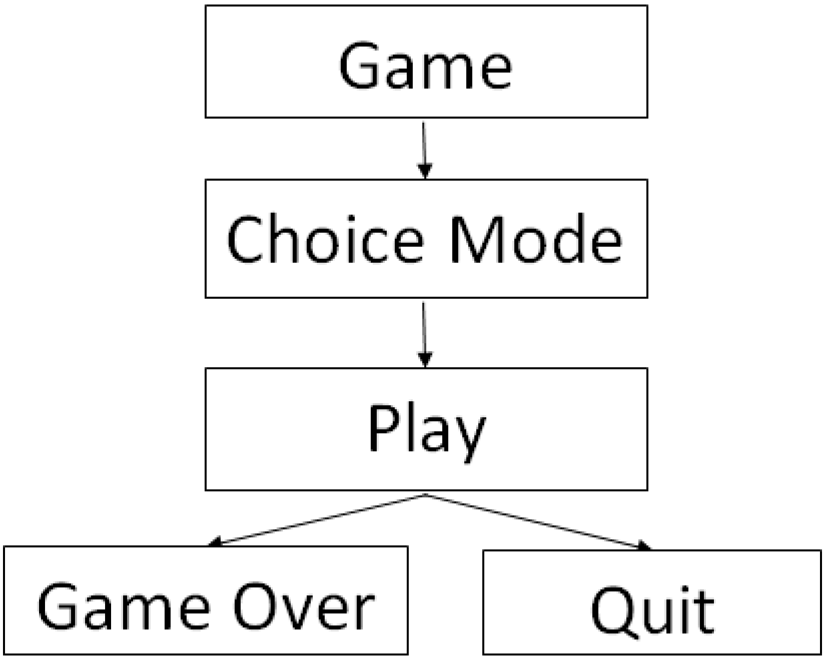
Introduction

Presentation of the project and key objectives



The Project is called Tetris Like Game. The aim is to create a game based on the Classic Tetris Game where the goal for the player is to make as many lines as possible with blocs. Now, we have to create new type of board : diamond, triangle and circle.

At the beggining, we propose three choices :
First : Begin a game, then he can return to the menu
Second : Read the rules, then he can return to the menu
Third : Quit the game



When we begin the game, the choice mode is launched:

First : the player chooses the shape of the board among triangle, diamond and circle.
Second : he can chooses between basic size and personnalized size.

Then, he can starts the game :
We print the board choosen and proposed to choose among three random block associated to this.
The player enter the coordinate of the block choseen. He try to increase the score by removing lines with the blocks.
He have three attemps to enter valid coordinate and then he loose.
However, each round, he have the possibility to quit the game.

This project allows us, to improve our skills in Basic notions, 1D lists, 2D lists, functions, modules and files.

Based functions

About Game Boards :

Functions	Input	Output
read_grid(path)	Relative naming	valid grid from the contents of the file specified by path
save_grid(path, grid)	path to save to grid and the grid	nothing
print_grid(grid)	grid	grid

About blocs:

Functions	Input	Output
print_blocs(grid)	shape of the grid	three blocs associated with it

select_bloc()	nothing	bloc choosen

About the position entered by the user :

Functions	Input	Output
valid_position(grid, bloc, i, j)	shape of the grid, bloc choosen, abcissa coordinate i, ordonate coordinate j	valid_position True or False (boolean)
emplace_bloc(grid, bloc, i, j)	shape of the grid, bloc choosen, coordinate i and j	the bloc choosen is inserted in the grid

About cancelling rows and columns, then calculating the score :

Functions	Input	Output
row_state(grid, i)	Shape of the grid, coordinate i	row_state True if the line choosen is full, else it is False (Boolean)
col_state(grid, j)	shape of the grid, coordinate j	col_state True if the column choosen is full, else it is False (Boolean)
row_clear(grid, i)	shape of the grid, coordinate i	new grid with the full line cleared
col_clear(grid, j)	shape of the grid, coordinate j	new grid with the full column cleared
update_score()	nothing	New score (integer)

New functions

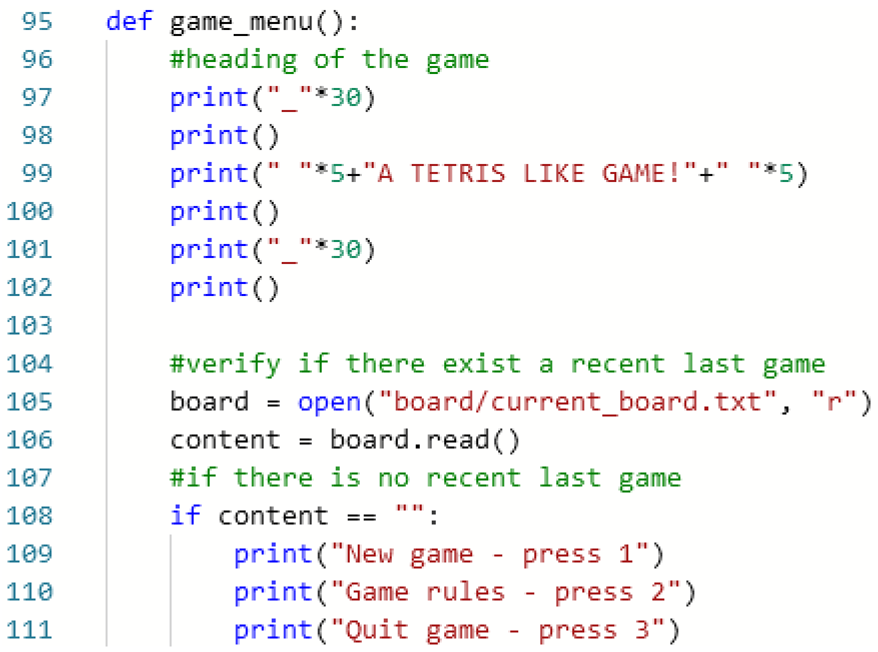
Functions	Input	Output
open_block(grid)	shape of the grid	read the blocs's files

Technical presentation of the project

Main implemented algorithms

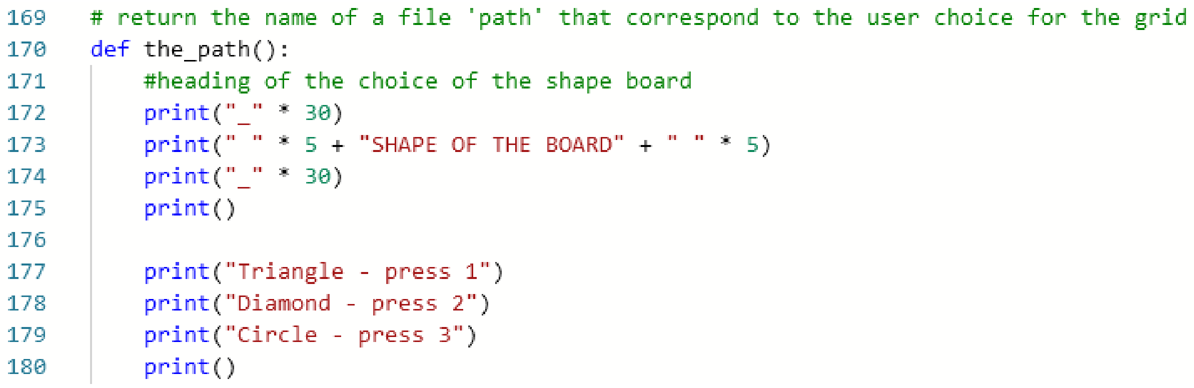
To play at this Tetris Like Game, the player have to run the program in main file.

The game part launch with three options : New Game, Game rules and Quit game



```python
def game_menu():
    #heading of the game
    print("_"*30)
    print()
    print(" "*5+"A TETRIS LIKE GAME!"+" "*5)
    print()
    print("_"*30)
    print()

    #verify if there exist a recent last game
    board = open("board/current_board.txt", "r")
    content = board.read()
    #if there is no recent last game
    if content == "":
        print("New game - press 1")
        print("Game rules - press 2")
        print("Quit game - press 3")
```

Then, we propose three board : Triangle, Diamond and Circle and after the size

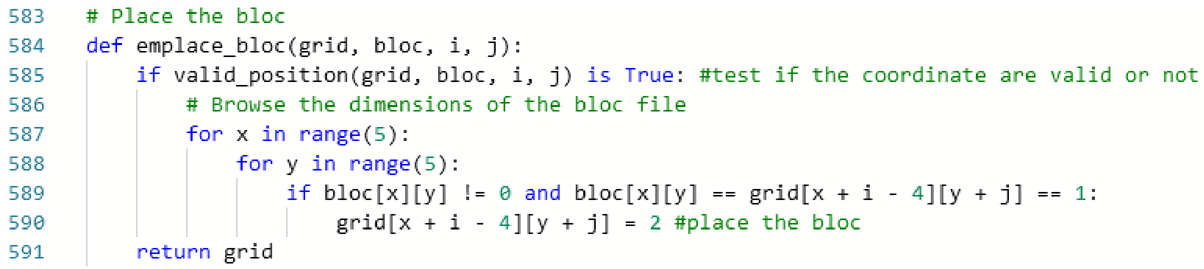


```python
# return the name of a file 'path' that correspond to the user choice for the grid
def the_path():
    #heading of the choice of the shape board
    print("_" * 30)
    print(" " * 5 + "SHAPE OF THE BOARD" + " " * 5)
    print("_" * 30)
    print()

    print("Triangle - press 1")
    print("Diamond - press 2")
    print("Circle - press 3")
    print()
```

After, we show three random blocs associated to the shape of the board with the function print_blocks(grid). The user choose a bloc among this list with the function select_bloc(grid) by entering for example block1 or block2 etc...

Subsequently, the user have three attemps to enter valid coordinate. To verify if the coordinate are valid, that is, it enter in the grid and the coordinates are not the same as another block already, we use the function valid_position(grid, bloc, i, j). This fonction return a boolean : True if the coordinate are valid or False if the coordinate are not valid.

If the coordinate are valid, we place the bloc into the board with the function emplace_bloc(grid, bloc, i, j)



```
# Place the bloc
def emplace_bloc(grid, bloc, i, j):
    if valid_position(grid, bloc, i, j) is True: #test if the coordinate are valid or not
        # Browse the dimensions of the bloc file
        for x in range(5):
            for y in range(5):
                if bloc[x][y] != 0 and bloc[x][y] == grid[x + i - 4][y + j] == 1:
                    grid[x + i - 4][y + j] = 2 #place the bloc
    return grid
```

The score increase each time a row or a column is clear.

Technical dificulties

The main crucial point was the personnalized board because we have to do all the changes per lines and columns in the all board. We had to do somes test to code this. There was also the problem of generate the circle board because for this, we had to do some mathematics with calculation about radius.

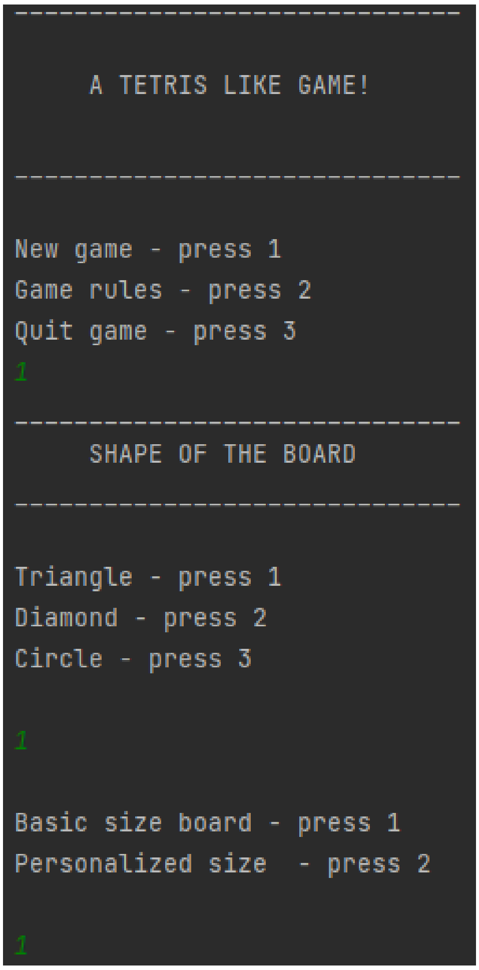
Another problem that we met is about the clear of a line. Indeed, when a clear a line, we have to do a translation of block above to the down direction whereas when the column is full state, we just have to clear it. We had to to some test to success.

We also try to code the function to rotate blocks and it was so difficult. Indeed, we tried to represent blocs on a paper with all the rotation possible but we didn't seeany logic between this rotation. We thought that it was a story of translations, but no. We do some research on internet to understand the logic and we didn't understand the solution, so we didn't code this function.

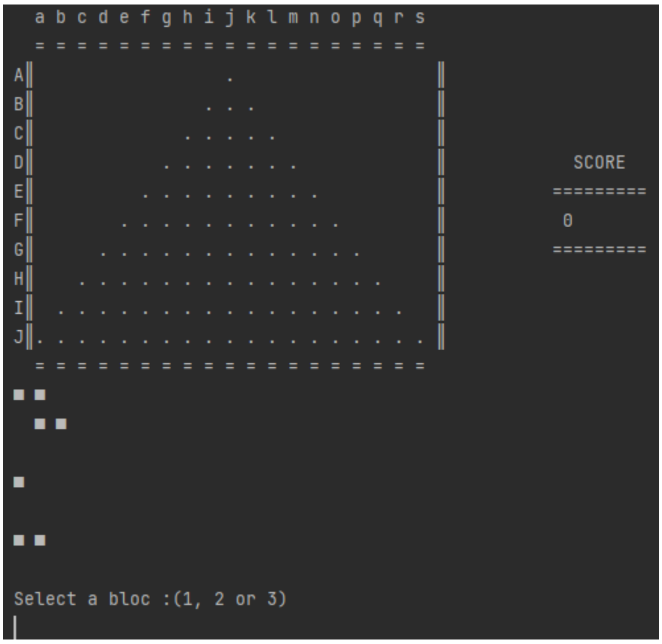
Result presentation

User interface

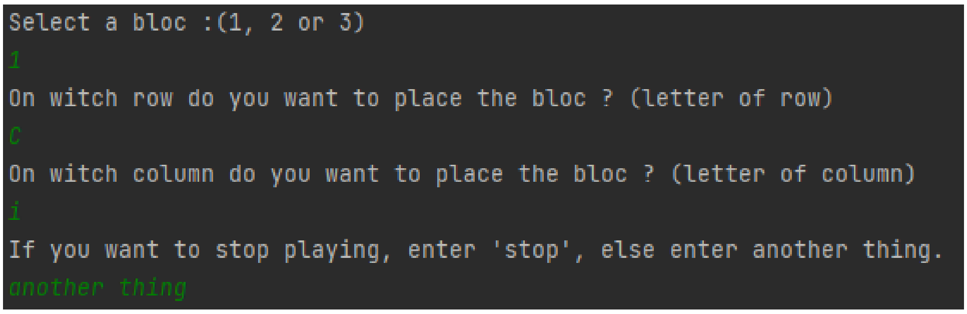
If we choose to play a new game with the triangle board and basic sized, then :



```
------------------------------

     A TETRIS LIKE GAME!


------------------------------

New game - press 1
Game rules - press 2
Quit game - press 3
1
------------------------------
     SHAPE OF THE BOARD

------------------------------

Triangle - press 1
Diamond - press 2
Circle - press 3

1

Basic size board - press 1
Personalized size  - press 2

1
```

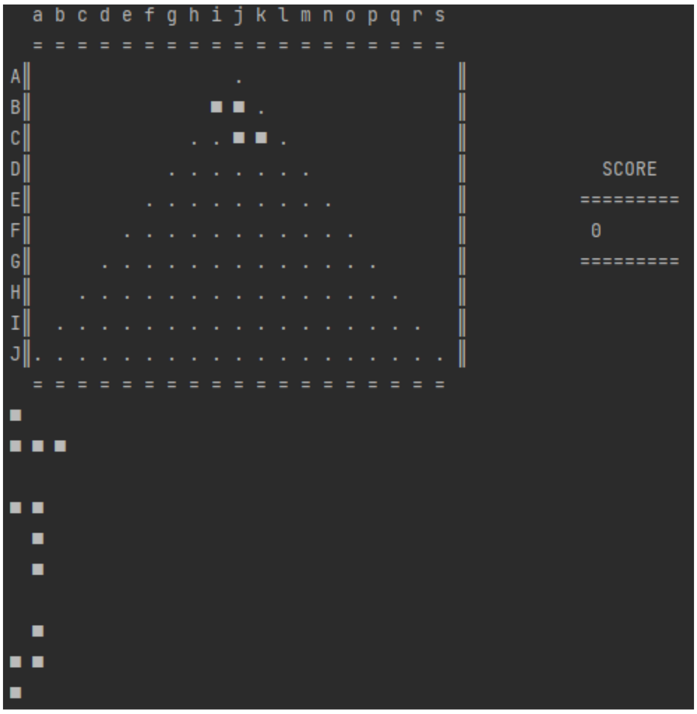


```
  a b c d e f g h i j k l m n o p q r s
  = = = = = = = = = = = = = = = = = = =
A║                  .                   ║
B║                . . .                 ║
C║              . . . . .               ║
D║            . . . . . . .             ║            SCORE
E║          . . . . . . . . .           ║          =========
F║        . . . . . . . . . . .         ║           0
G║      . . . . . . . . . . . . .       ║          =========
H║    . . . . . . . . . . . . . . .     ║
I║  . . . . . . . . . . . . . . . . .   ║
J║. . . . . . . . . . . . . . . . . . . ║
  = = = = = = = = = = = = = = = = = = =
■ ■
  ■ ■

■

■ ■

Select a bloc :(1, 2 or 3)
```



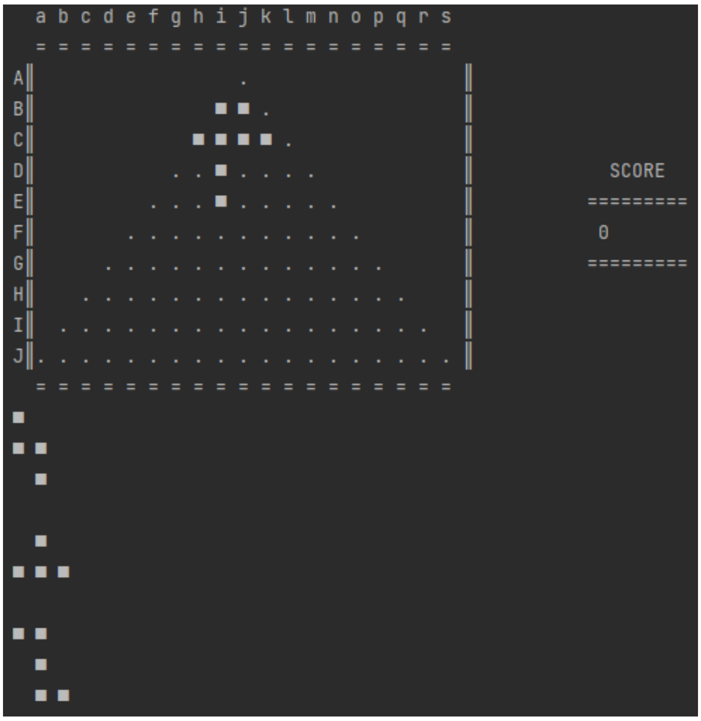
```
Select a bloc :(1, 2 or 3)
1
On witch row do you want to place the bloc ? (letter of row)
C
On witch column do you want to place the bloc ? (letter of column)
i
If you want to stop playing, enter 'stop', else enter another thing.
another thing
```

It place the bloc choosen into the board

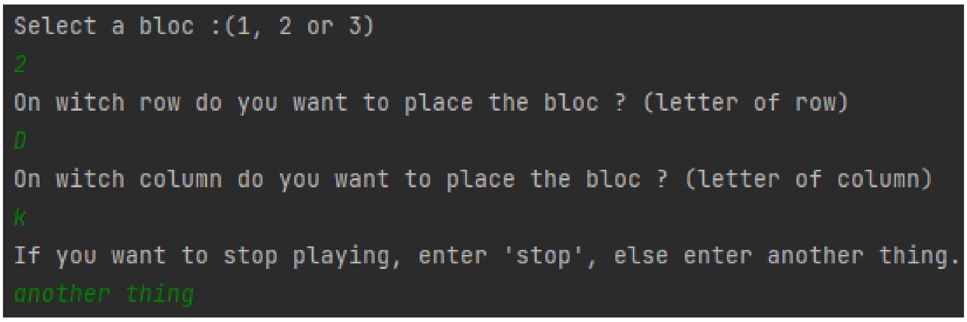


```
  a b c d e f g h i j k l m n o p q r s
  = = = = = = = = = = = = = = = = = = =
A║                  .                   ║
B║                ■ ■ .                 ║
C║              . . ■ ■ .               ║
D║            . . . . . . .             ║            SCORE
E║          . . . . . . . . .           ║          =========
F║        . . . . . . . . . . .         ║           0
G║      . . . . . . . . . . . . .       ║          =========
H║    . . . . . . . . . . . . . . .     ║
I║  . . . . . . . . . . . . . . . . .   ║
J║. . . . . . . . . . . . . . . . . . . ║
  = = = = = = = = = = = = = = = = = = =
■
■ ■ ■

■ ■
  ■
  ■

  ■
■ ■
■
```

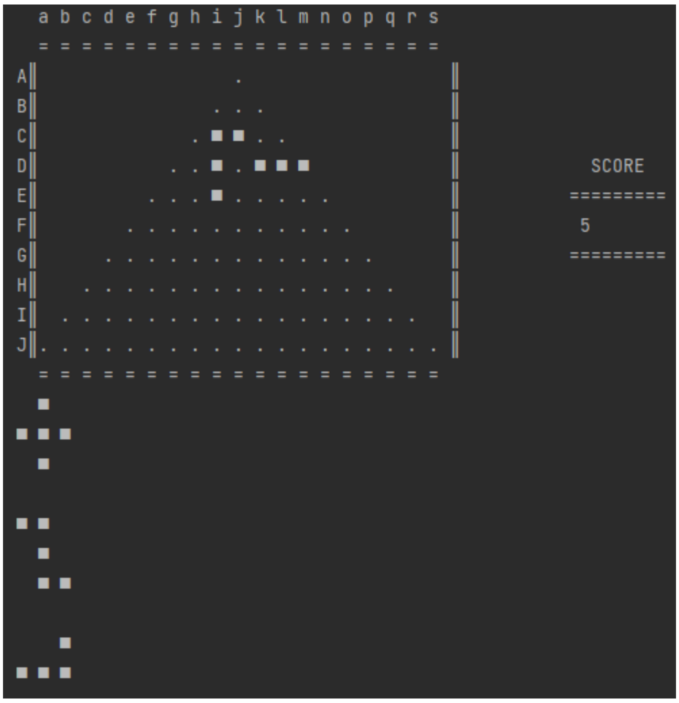
After a few rounds of play, we decide to clear a line , and the line above goes down:



```
  a b c d e f g h i j k l m n o p q r s
  = = = = = = = = = = = = = = = = = = =
A║                  .                   ║
B║                ■ ■ .                 ║
C║              ■ ■ ■ ■ .               ║
D║            . . ■ . . . .             ║            SCORE
E║          . . . ■ . . . . .           ║          =========
F║        . . . . . . . . . . .         ║           0
G║      . . . . . . . . . . . . .       ║          =========
H║    . . . . . . . . . . . . . . .     ║
I║  . . . . . . . . . . . . . . . . .   ║
J║. . . . . . . . . . . . . . . . . . . ║
  = = = = = = = = = = = = = = = = = = =
■
■ ■
  ■

  ■
■ ■ ■

■ ■
  ■
  ■ ■
```



```
Select a bloc :(1, 2 or 3)
2
On witch row do you want to place the bloc ? (letter of row)
D
On witch column do you want to place the bloc ? (letter of column)
k
If you want to stop playing, enter 'stop', else enter another thing.
another thing
```



```
  a b c d e f g h i j k l m n o p q r s
  = = = = = = = = = = = = = = = = = = =
A║                  .                   ║
B║                . . .                 ║
C║              . ■ ■ . .               ║
D║            . . ■ . ■ ■ ■             ║          SCORE
E║          . . . ■ . . . . .           ║        =========
F║        . . . . . . . . . . .         ║         5
G║      . . . . . . . . . . . . .       ║        =========
H║    . . . . . . . . . . . . . . .     ║
I║  . . . . . . . . . . . . . . . . .   ║
J║. . . . . . . . . . . . . . . . . . . ║
  = = = = = = = = = = = = = = = = = = =
  ■
■ ■ ■
  ■

■ ■
  ■
  ■ ■

    ■
■ ■ ■
```

Conclusion

This project made us realize that a big goal like creating a game involves breaking it down into small, interconnected problems.

We got down to it as soon as we were made aware of the project and this allowed us to list some of the problems that would take us more time.

Some functions required days of reflection and many tests in order to succeed.

In other cases, it was by communicating with our classmate that we were able to unblock situations and if not, we also sought to improve our functions by documenting ourselves on the internet. Many times, we did not immediately understand the solutions offered on the internet but we did not give up and we persevered to obtain an affordable result.

There are certainly improvements to be made, whether graphical or even to the core of our code, but at least we were able to have a concrete result that works.